[illegible]

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/944,685	08/31/2001	Henry Fang	SLA 1070	2111

7590 01/26/2005

David C. Ripma
Patent Counsel
Sharp Laboratories of America, Inc.
5750 NW Pacific Rim Boulevard
Camas, WA 98607

EXAMINER	
VO, TED T	
ART UNIT	**PAPER NUMBER**
2122	

DATE MAILED: 01/26/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Offic Action Summary	**Application No.** 09/944,685	**Applicant(s)** FANG, HENRY	
	Examiner Ted T. Vo	**Art Unit** 2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Peri d for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 09 September 2004.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-17 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-17 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the communication filed on 09/09/2004 responsive to the Office action dated, 06/29/04.

Claims 1, 4, 6-7, 9, and 12 are amended.

Claims 1-17 are pending in the application.

Claims 1-17 remain rejected under 35 U.S.C. 112, second paragraph, and Claims 1-17 stand finally reject under 35 U.S.C. 102(a) as being anticipated by HAVi SPECIFICATION.

Response to Arguments

2. Applicants' arguments in the Remarks section filed on 09/09/2004 have been fully considered.

- Regarding amendments/arguments to Claims under 35 U.S.C 101: In view of the amendment to the preamble of independent Claims 1, 7, and 12, "*In a signal bearing medium tangibly embodying a program of machine-readable instructions executed by an open source, interoperable IEEE 1394 specification digital device, a method for defining device user interface controls*", associating with "*retrieving virtual key information*", this makes a tangible method. The rejection under 35 U.S.C 101 is withdrawn.

- Regarding amendments/arguments to Claims under 35 U.S.C 112, second paragraph: In view of the amendment that amends to remove the Trademark subject mater, "HAVi Specification", the amendment with new recitation "*a method for defining device user interface controls*" remains rejected under 35 U.S.C. 112, second paragraph. See rationale in sections 3-4 below.

- Response to Applicants' arguments to Claim 1-17, rejected under 35 U.S.C. 102(a) as being anticipated by HAVi SPECIFICATION:

- - In the Remarks section, in page 8, at lines 1-3, Applications wrote, "In Section 7 of the Office Action claims 1-17 have been rejected under 35 U.S.C. 102(e) as being anticipated by HAVi Specification Version 1.1, 5-2001", it should be noted that the previous rejection of Claims 1-17 is under 35 U.S.C. 102(a).

- - In the Remarks section, started at line 16 of page 8 to line 9 of page 9, Applicants argue that all the elements participated in claims must be found, either expressed or inherently in a single reference" by asserting the facts that,

> HAVi Section 2.5.2 states that the L2 UI is based upon JAVA AWT 1.1. Section 7.4 states that JAVA code units are entities for uploading, and that the format of a JAVA code unit is the JAR format. Details are given of DCM, AMC, and Havlet code units. Section 8.1 describes the RAW UI. Section 8.1.1 notes that "a large number of events are optional, allowing manufacturers to customize and add value to their products." Section 8.3.2.4 states that there are three classes available to determine if a device is available. Section 8.7 describes a general approach to error behavior. Section 8.8 is a list of constants. Section 9.4 presents a list of HAVi key values. Section 9.5 lists HAVi and non-HAVi ROM requirements.
> Section 8.3.2.5 (page 429) of the HAVi specification states that an event can have a representation as a string, color, or symbol, which can be determined by calling getString, getColor, and getSymbol, respectively. This methodology permits the definition of a device button. However, HAW Section 8.3.2.5 does not teach how the virtual key information is to be stored or retrieved from memory.

Examiner respectfully responds: First of all, in the Applicants' specification, it admits the use of the prior art (specification: started in page 5, at line 11, to page 6, to line 6, including table 1), where the prior art is the HAVi SPECIFICATION device/protocol given by HAVi, Inc. All claimed elements such as Level 2, User interface, virtual key, JAR file, byte code, are the elements used in the HAVi Specification. Furthermore, the above Application's assertions tend to cut short, not to extend to the whole HAVi specification which incorporates the Claims' limitations. Respectfully, the whole consideration of HAVi Specification would be required.

HAVi discusses claimed features/elements: For example, in page 5, row 10, HAVi discloses HAVi Level 2 interoperability, "*Refers to the features provided by FVAs and Upload DCMs*" (Note: FVA: a HAVi-compliant device runs software elements of HAVi, including Java runtime; DCM: Device Control

Module). In page 22, third paragraph, HAVi discloses that "*Level 2 interoperability appears when one considers resources needed to access device functionality. Level 2 allows a device to be controlled via an upload uploaded DCM that presents all the capabilities offered by the device*". These discussing features particularly disclose, "*from a level two (L2) graphical user interface (GUI)*".

In page 24, section 2.9.2, HAVi discloses, "*Uploadable DCMs, uploadable Application Modules and all havlets* ***are obtained from*** *"code units" – these* ***are JAR files***". In section 8.3.2.5 (page 429), HAVi discloses Virtual keys information is such as VK_COLORED_KEY_0,,, VK_COLORED_KEY_5, etc. And it is understandable that these keys are held together by getColor [Note: VK stands for Virtual Key: these VK_...keys are the bundled in accordance to getColor]. These discussing features particularly disclose Claim 1 (and the similar manner to Claim 7 and 12), "*from a level two (L2) graphical user interface (GUI), accessing a JAR file; and in response to accessing the JAR file, retrieving virtual key information*".

The whole HAVi Specification shows that the accessibility (and thus retrieval) of code in the memory is via the Level 2 UI. A user acts the accessibility and the information retrieval of the code in the memory including the code of JAR file, Java input/output ResourceBundle, Java byte code. The elements of the code includes virtual key information.

- - With regard to the argument to Claim 1 that HAVi 7.4 and 8.3.2.5 do not teach the access of virtual key information from a JAR file (Remarks, page 9, lines 10-12).

Examiner disagrees. The whole reference should be referred. As discussed and pointed by Examiner above, JAR files are discussed in the HAVi Specification and containing virtual key information (VK_COLORED_KEY_0, VK_COLORED_KEY_5, etc.), where the Level 2 UI of HAVi supports a user to access and obtain code of class files or JAR file (See Definition HAVi Level 2 in page 5, and in section 2.5.2, and See section 7.2.2, page 394 for having DCMs and accessing the JAR files). Figure 25, page 89, illustrates such key obtaining.

- - With regard to the argument to Claim 7 that HAVi 8.3.2.4 and 8.3.2.5 do not teach the retrieval of Virtual key information in response to accessing a ResourceBundle (Remarks, page 9, lines 12-15).

Examiner disagrees. Claim 7 is clearly having the same claimed functionality as Claim 1. As discussed and pointed out by Examiner above, HAVi specification provides a user to access class files and JAR files which include virtual key information. With this interfacing, the user can modify, get, obtain, save, store, (meaning retrieve) or do whatever the user wants. The ResourceBundle as given in the Applicants' specification is a set of input/output representations in the class file/Jar files. HAVi 8.3.2.4 and 8.3.2.5 discuss about the term, "input elements" such as String, Color, Symbol, etc. For example, with color, it has 6 keys, VK_COLORED_0,,, and VK_COLORED_5 (ResourceBundle), and these keys are obtained (retrieved) from calling "getColor" (See texts in section 8.3.2.5, and also see page 24, section 2.9.2).

- - With regard to the argument to Claim 12 that HAVi 7.4, 8.7, 8.8, 9.4, 9.5, and 8.3.2.5 do not teach the use of a JNI to access mapped memory, to retrieve virtual key information (Remarks, page 9, lines 15-19).

Examiner disagrees. Respectfully, the whole HAVi specification would be required for this argument. As discussed and pointed out by Examiner all above, HAVi specification provides the user to access the keys with the use of Level 2 UI to the code like the code of JAR file/Class file, where the code always resides in the memory. HAVi discusses code unit in section 7.4 (pages 414-415). In this section, HAVi states that the format of a Java Code Unit is the Java achieve or "JAR" format". Also referring to the definition of code unit, page 4, row 5, "Code Unit", it states, "*Native code units are platform dependent and typically would include machine code for a specific processor. Java code units include Java byte code*", and see description of "native code unit" in page 5, row 20. In page 89, HAVi shows uploading/obtaining keys within secure storage ('*mapped memory*') from the root key of the FAV (FAV as discussed above is Java runtime device). Interfacing to Code units thus has means of Java native interface (JNI). With these discussions, HAVi discloses the limitation of Claim 12. Furthermore, in chapter 9, section 9.4 and 9.5 (pages 454-455), it shows the tables with Keys' representations and the ID values, and further in the sections 9.7, discussing configuring ROM, 9.8, root directory, all these have means for accessing mapped memory where in this memory JAR file native code is resided.

Applicant's arguments are not persuasive. The rejections of Claims 1-17 stand finally rejected as set forth below.

Claims 1-17, previously rejected under 35 U.S.C 112, second paragraph, and under 35 U.S.C. 102(a) as being anticipated by HAVi SPECIFICATION Version 1.1, stand finally rejected in this action.

Claim Rejections - 35 USC § 112

3. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

4. Claims 1-17 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 1, 7, and 12: Claims 1, 7, and 12 which recite newly limitation "*defining device user interface controls*" in response to the prior rejection do not make the scopes of the Claims clear. The recitation in these Claims' preambles is not clear in combining with the step's functionality in the claim body, "*retrieving virtual key information*". There are lacking essential elements or features in the claim to connect to the preamble, particularly the preamble recitation "*defining device user interface controls*". It should be noted that the preamble of a claim sets forth what the claim is. The functionality of each Claim 1, 7, or 12, particularly, "*retrieving virtual key information*", does is not reflect what it is set forth by the claim preamble. This causes the claim fail to particularly point out and distinctly claim the subject matter. It would suggest amending the claimed preambles of these Claims to consist and to reflect to the body of the Claims.

Claims 2-6, 8-11, 13-17: Claims are dependent on the Claims that are identified as being indefinite above. Therefore these Claims are indefinite based on its dependency.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

6. Claim 1-17 are rejected under 35 U.S.C. 102(a) as being anticipated by HAVi SPECIFICATION Version 1.1, 5-2001 (hereinafter: HAVi).

Given the broadest reasonable interpretation of followed claims in light of the specification.

As per claim 1: HAVi discloses,

In a signal bearing medium tangibly embodying a program of machine-readable instructions executed by an open source, interoperable IEEE 1394 specification digital device, a method for defining device user interface controls, the method comprising:

from a level two (L2) graphical user interface (GUI), accessing a JAR file; (See page 5, row 10, 'Level 2 interoperability', in page 22, third paragraph, 'access device functionality', page 18, section 2.5.2, Level 2 UI, page 24, section 2.9.2, 'are JAR files'); *and,*

in response to accessing the JAR file, retrieving virtual key information (See page 24, section 2.9.2, 'obtained from "code unit", page 429, section 8.3.2.5, User Input Representation, 'getString', 'getColor', 'getSymbol', 'VK_COLORED_KEY_0',,, 'VK_COLORED_KEY_5', *retrieving virtual key information*).

As per claim 2: HAVi discloses, *The method of claim 1 wherein accessing a JAR file includes accessing a JAR file stored in read only memory (ROM)* (See page 455, section 9.5,Table 18, HAVi Configuration ROM Requirement).

As per claim 3: HAVi discloses, *The method of claim 2 wherein retrieving virtual key information includes retrieving virtual key information from a JAR file model selected from the group including static classes and data arrays.* (See page 90, section 3.10.2, Figure 26) by showing a code unit of a Jar file that includes *static classes and data arrays.*

As per claim 4: HAVi discloses, The *method of claim 3 wherein retrieving virtual key information in response to accessing the JAR file includes retrieving an application bundled with the virtual key information* (See page 429, section 8.3.2.5, User Input Representation, "by calling "getString", "getColor", "getSymbol", *bundled with the virtual key information*).

As per claim 5: HAVi discloses, *The method of claim 4 in which a first microprocessor machine using a first operating system is included; the method further comprising: receiving virtual key information as Java source code; using a Java compiler, compiling the Java source code into Java virtual machine (JVM) byte codes for the first operating system; and, using jar tools, archiving the JVM byte codes into a JAR file stored in ROM* (See Page 395, Java 1.1 Core API, and Jave.lang.Compiler).

As per claim 6: HAVi discloses, The *method of claim 5 further comprising: receiving the application as Java source code; using a Java compiler, compiling the Java source code into Java virtual machine (JVM) byte codes for the first operating system; and, using jar tools, archiving the JVM byte codes into a JAR file stored in ROM* (See Page 395, Java 1.1 Core API, and Jave.lang.Compiler).

As per claim 7: HAVi discloses,

In a signal bearing medium tangibly embodying a program of machine-readable instructions executed by an open source, interoperable IEEE 1394 specification digital device, a method for defining device user interface controls, the method comprising:

from a level two (L2) graphical user interface (GUI) accessing a Java input/output (I/O) ResourceBundle (See page 5, row 10, 'Level 2 interoperability', in page 22, third paragraph, 'access device functionality', page 18, section 2.5.2, Level 2 UI, page 24, section 2.9.2, 'are JAR files', see page 429, section 8.3.2.4, User Input Capabilities, "getInputDeviceSupport", section 8.3.2.5, User Input Representation, "Org.havi.ui.event.HrcCapabilities class" that contains getRepresentration); *and,* *in response to accessing the ResourceBundle, retrieving virtual key information* (See page 24, section 2.9.2, 'obtained from "code unit", page 429, section 8.3.2.5, User Input Representation, 'getString', 'getColor', 'getSymbol', 'VK_COLORED_KEY_0',,, 'VK_COLORED_KEY_5', *retrieving virtual key information*).

As per claim 8: HAVi discloses, *The method of claim 7 wherein accessing the ResourceBundle includes using a ResourceBundle application program interface (API) to specify a property file* (Referring to Level 2 UI, page 425; where Level 2 UI is related to API, and see page 429, section 8.3.2.5, User Input Representation, referring to class that defines an event having a known representation; For example six colored key events).

As per claim 9: HAVi discloses, *The method of claim 8 in which a first microprocessor machine using a first operating system is included; the method further comprising: maintaining an application in a protocol associated with the first operating system; and, wherein accessing the ResourceBundle includes using a ResourceBundle API to specify a property file stored in a file system associated with the first microprocessor machine* (See pages 414-419, section 7.4 Code Units (for install and uninstall Jar files));

As per claim 10: HAVi discloses, *The method of claim 9 wherein using a ResourceBundle API to specify a property file stored in the file system includes specifying a property file stored in an input/output (I/O) device selected from the group of storage devices including hard disks and Flash memory* (See page 6, section 1.4 referring to "persistent memory" for storage devices).

As per claim 11: HAVi discloses, *The method of claim 10 further comprising: receiving virtual key information as text-based properties attributes in a ResourceBundle property file; integrating the virtual key information into a table of virtual key characteristics; and, storing the virtual key characteristics table as the ResourceBundle property file* (See page 429, section 8.3.2.5, User Input Representation, referring to the representation getString that allows a text representation).

As per claim 12: HAVi discloses,

In a signal bearing medium tangibly embodying a program of machine-readable instructions executed by an open source, interoperable IEEE 1394 specification digital device, a method for defining device user interface controls, the method comprising:

from a level two (L2) graphical user interface (GUI) calling a Java native interface (JNI); at the JNI, using Java byte codes to call a storage driver (See page 5, row 10, 'Level 2 interoperability', in page 22, third paragraph, 'access device functionality', page 18, section 2.5.2, Level 2 UI, page 24, section 2.9.2, 'are JAR files', and see pages 414-419, section 7.4 Code Units; see section 8.1 in page 425, HAVi level two UI allows applications written in Java; and see pages 448-453, section 8.7, and 8.8; and see page 89, Figure 25)*;*

from the storage driver, accessing a mapped memory (See page 89, Figure 25, see section 9.4 and 9.5, pages 454-455, HAVi Unit Directory)*; and, in response to accessing the mapped memory, retrieving virtual key information* (See page 89, Figure 25, and see page 429, section 8.3.2.5, User Input Representation, "getString", "getColor", "getSymbol", *retrieving virtual key information*).

<u>As per claim 13</u>: *The method of claim 12 wherein accessing a mapped memory includes accessing a mapped memory stored in an electrically erasable programmable read only memory (EEPROM)* (See page 55, Figure 18, code unit; and see page 455, section 9.5,Table 18, HAVi Configuration ROM Requirement).

<u>As per claim 14</u>: *The method of claim 13 wherein retrieving virtual key information includes retrieving virtual key information from mapped memory in a binary format* (See section 9.11.1, pages 429-430, for Keys that represents virtual key information in numbers).

<u>As per claim 15</u>: *The method of claim 14 wherein using Java byte codes to call a storage driver at the JNI includes converting the Java byte code to binary format addresses (inherent in JVM); and, wherein accessing a mapped memory from the storage driver includes using the binary format addresses to access ASCII codes stored in the EEPROM* (See section 9.11.1, pages 429-430, for Keys that represents virtual key information in numbers).

<u>As per claim 16</u>: *The method of claim 15 in which a first microprocessor using a first operating system is included; the method further comprising: receiving the storage driver as first operating system machine codes; and, storing the storage driver as machine code* (See page 54, last two paragraphs, see page 55, Figure 18, and third paragraph).

As per claim 17: *The method of claim 16 further comprising: receiving virtual key information as binary format code; using the storage driver, cross-referencing the virtual key information with EEPROM addresses; and, storing the virtual key information in the EEPROM as machine code* (See page 54, last two paragraphs, see page 55, Figure 18, and third paragraph).

Conclusion

7. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 5:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3694. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]

Ted T. Vo
Primary Examiner
Art Unit 2122
January 14, 2005